VOLUME III, NO. 5 -:- MAY 1981

OSIO Newsletter

Pascal, the Continuing Saga

By David Morganstein

In this installment, I will talk about the Pascal system's limits on source code size, and some ways to get around these limitations. Included is a program called SPLITEXT which can be used to deal with the problem.

To begin, I will mention an organization which UCSD users should consider joining: the UCSD p-System Users' Society, USUS to its members. USUS costs twenty dollars annually, and provides a newsletter and library to its members. I joined recently and sent for their five 8" library disks in OSI floppy format. The library contains many useful (and some useless) programs. For the beginner, example programs to modify and personalize are a great starting point. These disks are a good buy for the sixty-dollar price tag.

The most interesting-looking programs in the USUS library were a text processor and a data base manager. However, all was not that easy. These programs were very large, so large that the pieces provided would not fit in the 10K of space provided in our somewhat shrunken OSI version of UCSD Pascal. What to do??? Well, the ideal solution would be to further break up the text files provided. How to do this when you can not get a piece of code into the editor?

As an aside, I mention that the editor does allow you to copy pieces of code into the editor workspace. But these pieces must be defined by previously installed markers. Since I had not written these chunks of code, and whoever did write them had not installed any such markers, this option was not available.

The program below, SPLITEXT, was printed in the San Francisco Cider Press, a publication of a competing computer user group, the name of which I will not deign to include in our hallowed pages.... however, it does pay to see how the other half lives. Anyway, SPLITEXT filled the bill. You see, Pascal source code comes in plain old text files, which can be read

continued on page 2

OS-65D Notes

By Joe Kirshner

THE DISK DIRECTORY AND ASSOCIATED ITEMS

I received a disk from Paul Rainey this week containing a couple of interesting programs. One of these is a procedure for changing the order of the titles in the directory. This allows you to group particular titles together or to sort the titles once and then replace them sorted so that you do not have to wait for the directory to sort them every time you want to look at them.

The second program zeroes out the garbage in the random access files.

One of the techniques used by Paul in his program to change the directory is that of passing a parameter between two programs. As I mentioned in one of the previous articles, information can be passed from one program to another by writing it to device #5 (memory outside of BASIC's control) from one program, and then calling it back from your second program. The problem that I pointed out, however, was that if you try to call something from device #5 and there's nothing there, the computer will usually hang up. Paul writes the parameter to memory outside of BASIC by using POKEs and then reads it into the other program with PEEKs. Since peeking at an address isn't affected by the presence or absence of a carriage return, hang up of the computer will not occur even though you have not written anything to the address being examined.

You can, of course, write to memory either by using POKES or Device #5 and still look at it with PEEKs. However if you POKE something into memory, you will not be able to access it using Device #5 unless you have placed carriage returns in it.

Of course you must lower the top of memory to leave enough room for whatever you intend to store there. In this connection I have a note from Geoff Cohen (who is a member in Australia) that points out that the amount of memory available can be found by PEEKing at 8960

continued on page 2

continued from page 1

just like any other text file. Thus, you can read the first hundred or so lines and write them to one file, repeating the process until the original source file has been broken up into workable chunks.

Great, you say, now how do you get it all back together again to get it compiled?

No problem!!! The Pascal compiler recognizes a number of useful directives; special instructions provided right in the source code and acted upon during compilation. These directives are placed in comment delimiters, e.g. { } or (* *). The particular directive needed to pull together pieces of text at compile time is $IFILENAME. When it encounters this directive, the compiler searches the appropriate volume directory for the mentioned text file and uses its contents in the compiling process. As a matter of fact, the word processor and data base packages in the USUS library both work this way. One small shell program for each contains a series of {$IFILENAME} commands specifying the pieces.

To join USUS, send your twenty dollars to:

U.S.U.S.
Box 1148
La Jolla, CA 92038

```
Program SPLITEXT;

§written by STEVE LLOYD, San Francisco
 Apple Core†

var INFILE,OUTFILE: text;
    INNAME, OUTNAME, LINEBUF: string;
    MAXLINES, LINECOUNT: integer;

begin

writeln;
write('name of input file');
readln(INNAME);
if pos('.text',INNAME)=0
   then INNAME:=concat(INNAME,'.text');
(*$I-*)
reset(INFILE,INNAME);
if ioresult<>0 then
   begin
   writeln('file not found');
   exit(program)
   end;
(*$I+*)
while (not eof(INFILE)) do
   begin
   writeln;
   write('name of output file: ');
   readln(OUTNAME);
   if pos('.text',OUTNAME)=0
      then OUTNAME:=concat(OUTNAME,'.text');
   rewrite(OUTFILE,OUTNAME);
   write('maximum number of lines: ');
   readln(MAXLINES);
   LINECOUNT:=0;

   repeat
     readln(INFILE,LINEBUF);
     writeln(OUTFILE,LINEBUF);
     LINECOUNT:=LINECOUNT+1;
     if(LINECOUNT MOD 10)=1 then write('.');
  until (eof(INFILE) or eof(OUTFILE) or
  (LINECOUNT=MAXLINES));
   writeln;
   writeln(LINECOUNT,' lines written
   to ',OUTNAME);
   if eof(OUTFILE)
      then begin
      writeln('not enough room in output
       file');
      close(OUTFILE,PURGE)
      end
         else close(OUTFILE,LOCK);
      end;
      close(INFILE);
      end.
```

OS65-D Notes

continued from page 1

(decimal). This location holds the high byte. The low byte is zero.

```
(LINE #)INPUT"HOW MANY PAGES DO YOU WANT ABOVE BASIC";P
(LINE #)E=PEEK(8960)+1:S=E-P:POKE 132,0:POKE 133,S
```

Geoff writes that he uses this to put his buffers on the top. He writes the following lines as part of BEXEC*

```
(LINE #) P=8:REM  P=8 FOR 5IN. AND P=12 FOR 8IN. DISKS
(LINE #) E=PEEK(8960)+1:S=E-P:POKE 132,0:POKE 133,S
(LINE #) POKE 8998,0:POKE 8999,S
(LINE #) POKE 9000,0:POKE 9001,E
```

If you add the above lines to BEXEC*, you no longer need to bother about doing something about a buffer when you write a program that needs one since

Buffer #6 will always be available above the program. Of course this means that the top 12 (8 for Minis) pages of memory are not available even if you don't need a buffer in some particular program, but in those rare cases where you have a long program that doesn't require a buffer and needs all of your available memory, you can raise the top of memory in the first couple of lines of that program.

Getting back to passing parameters, Paul Rainey uses it in traveling back and forth between his RENAME program and the DIRECTORY. The parameter is used to set a flag (an indicator) to tell one program whether or not to RUN the other program.

If you have a program that is so long that it must be brought into memory a piece at a time, it may be necessary to start a program element from some point other than its beginning, where the line at which it should start is determined by the previous program element.

The second program would start with:

```
(LINE #) X=PEEK(ADDRESS OF PARAMETER)
(LINE #) ON X GOTO A, B, C, D
```

In the above A, B, C, & D are the line numbers of your program, and the parameter is a number from 1 to 4. The value of the parameter will, of course, have been written to the address in outer memory by the previous program segment.

Paul Rainey's programs will be available on the exchange disk and rather than taking up a great deal of space in the Newsletter, I think it will be more worthwhile to write up a number of demonstration programs illustrating these techniques and to add them to the exchange library.

Unfortunately, it will be at least a month until I will have the time to do so, but I will prepare number of programs and select some of the programs I already have for inclusion in the 65D exchange disk.

I intend to include the following:

Examples of the PEEKs and POKEs previously discussed.

A program for reading directories from each of a number of disks, sorting them, and then printing out a superdirectory using only a single disk system. The program will also inventory the number of empty tracks available on each disk.

An example of binary searching.

An example of the use of two buffers for long files, together with examples of insert and merge sorting.

RANDOM ACCESS FILES

OS-65D has random access files that allow you to put records keyed by numbers out to the disk and to access them directly even though the number of records is much too large to fit into your internal memory. Of course the main problem with such a system is that most of the time you don't want to call the records by number but rather by name so that some method is required for converting the name into a number before accessing the record. Random access records are rather difficult to print out in sorted order when that is desired, which is a second disadvantage of random access files.

On the other hand if you try to use sequential files for very large numbers of records, say 9 or 10 tracks, then you will find that searching for a particular record using a sequential search may take minutes. However, it is possible to set up the files to use a binary search that will find the record in a few seconds.

Nevertheless, random access is faster, and if you need to access only by number, it is already conveniently set up. As set up for an 8in. disk each track is broken into 24 sectors of 128 bytes each. If you want to change the record size from 128, you can change it to any power of 2 as follows:

```
(LINE #) POKE 12042,3072/N:POKE 12076, X
```

In the above N is the desired record size (128, 64, 32, ...) and X is the number such that 2©X=N, e.g. if N=64 then X=6

For a 5in. disk, use 2048 instead of 3072.

The DISK OPEN,6,"FILE#" command resets the addresses (12042 and 12076) to 128 bytes per record, consequently, the above line must be inserted after the DISK OPEN commands both when inputting from and outputting to the disk.

JOE KIRSHNER

Switch and Boot with Disk B

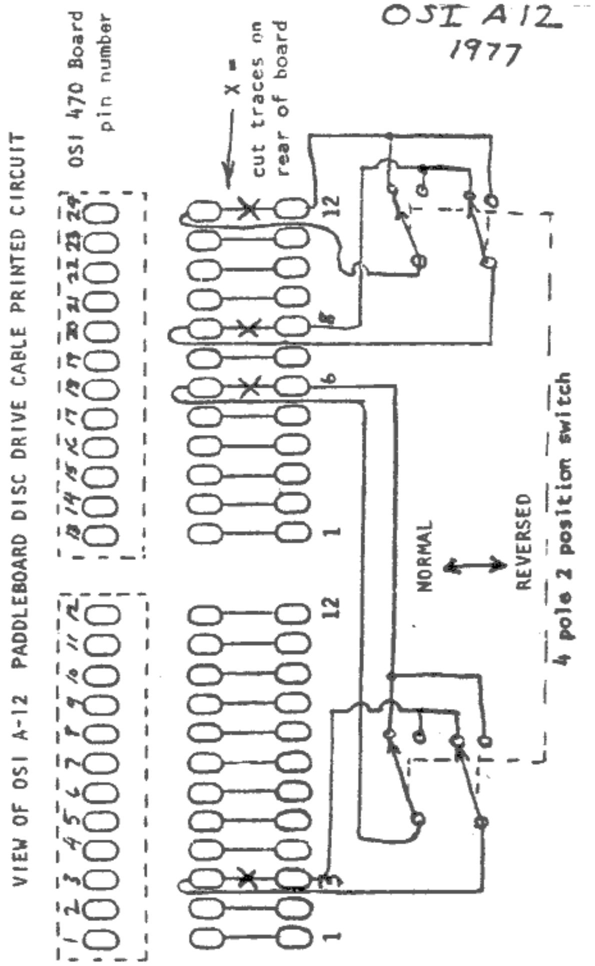
By Don Valentine, 450 7th Ave., NY 10123

Anyone who has used a dual disk system has probably gone through the frustrating experience of having the system fail to boot up properly. Trying to pin down the cause can be confusing -- is it the disk, the drives, the processor board, or memory?

Whatever the cause, trouble-shooting can be greatly simplified by a switch that reverses the system designation of the drives. Throw the switch and the top drive becomes Device B, the bottom one Device A. If the system boots up now, you're in business, at least with a one-drive system, and you can hang on until someone brings a screwdriver and a soldering iron.

If it doesn't, at least you've eliminated the disk unit from the list of most probable suspects. There are many other ways the switch can be used.

The diagram for wiring the switch is shown below. It simply criss-crosses or exchanges the "ready" and "select" lines to the two drives. The simplest place to do this is on the small paddle board attached to the drive ribbon cable, which plugs into the Molex connector for the OSI 505 or 470 board.



Use 12-inch wires to attach the switch to the paddle board, and you can mount the switch in one of the spare 1/2" holes in the rear of the C2- or C3-8P chassis.

Note that four traces are cut on the paddle board: F-3, Drive #1 select; F-18, Drive #2 select; F-20, Drive #2 ready; F-24, Drive #1 ready.

The time to do this mod is before problems arise.

DON VALENTINE

Members Notes

Dr. Jeffrey Krauss has sent us a list of practically all the Computer Bulletin Board systems operating around the country. We'd like to publish it, but it would take most of one issue, so we'd better not. We'll give a few of the numbers for the area where we have a high concentration of members, and precede that short list with a few observations about CBBS's:

(1) The things are multiplying like dandelions, so any list is incomplete before it's published. (Item: The one in Annapolis wasn't on Dr. Krauss's list - we added it.)

(2) Their popularity suggests that we're ready for some kind (or maybe more than one kind) of national data exchange system.

(3) If you have a modem and aren't using it much, you may be missing something useful and interesting.

(4) If you don't have a working modem, maybe it's time to start working on one, so you can get in touch with the following, among many, many others:

CBBS	Alexandria, VA	(703) 620-4990
CBBS	Washington, DC	(703) 620-4990
CBBS	Washington, DC	(202) 281-2125
CBBS	Washington, DC	(202) 337-4895
CP/M	Baltimore, MD	(301) 337-8825
CP/M	Baltimore, MD	(301) 655-0393
DBMS	Annapolis, MD	(301) 363-4867

Any CBB may be down from time to time, of course; call again later if you don't connect the first time.

Washington (2125) specializes in amateur radio.

DBMS Annapolis is on the air from 5:30 p.m. to 7 a.m.; most of the others operate 24 hours a day.

TAILGATE MARKET

MODEL 35 TELETYPE RO PRINTER -- Good condition heavy-duty printer mechanism removed from operating equipment. Reliable low-cost hard copy, sans case. $50. Complete set of 3 manuals $10. Chuck Popenoe, 7307 Wiscasset Road, Washington, DC 20016. Tel. (301) 229-0549.

THE SURVEY

Joe Kirshner has analyzed the questionnaire that was a part of the ballots in our recent election. To the question about what kind of Newsletter you would like to see in the future, here is how you rated the ideas that were listed, on a scale from 1 to 5:

4.1 User reports on systems, equipment, books, software packages, languages, etc.

3.53 Tutorial articles on assembly language and use of the assembler, the extended monitor, etc.

3.41 Programs, applications, and data on disk-based 6502 systems.

3.41 Programs, applications, and data on 65U systems.

2.45 Information about other available systems (FORTH, XPLO, etc.)

2.21 Articles on hard disk, multi-user, or network systems.

1.8 Programs, applications, and data on OS-CP/M.

1.4 Programs, applications, and data on BASIC-in-ROM or cassette systems.

You voted about 3 to 2 to have officers elected by the members instead of by the Board of Directors.

You were almost solid in your desire for a directory of members. Only one person said no, and not a single one said he would prefer not to have his name included. We'll start work now to get the information for the directory.

Many of you accepted our invitation to make other comments and suggestions. The officers will be studying those notes.

WHO'S MATRIX?

We've had several inquiries about the MATRIX in the return address of the Newsletter. That's the name of the firm which handles the printing and mailing for us. It's owned by George Marton, who is a founding member of OSIO and a Director.

For OSI business, you'll save time by using the address of the appropriate officer (the list is printed each month).

Membership Information

OSIO dues are $15 a year. Overseas members who want communications sent by airmail pay an additional $15, a total of $30 U.S.

If you are a member, your membership number and the date to which your dues are paid are shown above your name on the mailing label. Watch your label, and renew your membership promptly before it expires! We do not have the resources to mail renewal and reminder notices.

If no membership number is shown, this is a sample copy. We invite you to become a member, take part in our activities, and enjoy the benefits of our Newsletter, Program Exchange, chapter meetings, seminars, and occasional discount offers and group purchases. Just send a check in the proper amount to our Treasurer, whose name and address are given below.

Program Exchange

Our exchange disks do not contain the operating systems or other proprietary programs; add your own system to them.

Send in your own programs with enough postage to return the material to you, and if your program is accepted for the Exchange, we'll add other programs in the Exchange to your disk or cassette and return it.

Members who are not yet ready to provide their own programs to other members may get the Exchange programs by sending $8.50

to cover an Exchange disk and postage, or $3.50 for a cassette and postage. If you order a disk, remember to say whether 8" or 5-1/4", and whether 65U or or 65D. Please be sure to give your membership number.

A list of the programs available in the Exchange was in the Newsletter for September 1980. More have been added since then; we're working on a new list, and hope to have it ready next month.

Newsletter Back Issues

The Newsletter started with April 1979. The first few issues were short, but we tried to include some interesting, often useful routine or tip in each one. The early issues (to January 1980) are out of print; we have photocopied from them the items that are still of interest, omitting the outdated announcements, organizational notes, etc. If you'd like a complete set, we'll assemble one and send it to you for $6.50 plus postage. (The package weighs just over a pound, and goes by fourth class mail. Use your postal zone to estimate the cost of postage).

Be sure to give your membership number.

C1P Tyro-Typo

This is a simple program to turn the C1P into a practice typewriter to facilitate learning the touch system. If you just type away on a C1P, you get an error when you hit the return. This is distracting to the novice typist who is making enough honest mistakes.

The simplest approach is merely one line.

```
10 INPUT K$:GO TO 10
```

This however will trip over commas and the backspace still does not remove the unwanted character from the screen. The comma can be beat by starting off with " as the first character typed. There is a better way.

Here is a program that solves the problems. It still only types in upper case but that could be cured if you wish. I haven't programed in the "10 Easy Lessions to Touch Typing";that is easy enough to get out of a book.

The key to this program is that the keyboard routine at $FD00 (d.253-00) returns the ASCII value of the key pressed to d.530. We use USR(X)to jump to $FD00 and then look at the character found. If it is a 27 (ESC) we get out of the loop. If it is a 13 (return) we add a line feed. If it is a 127 (rubout) or95 (shift 'o')°backspace°, we jump to the video erase routine. If it is a 111 (shift rubout), we do a simple screen erase. If the chracter is anything else we print it on the screen.

The video erase works because the relative cursor positon is stored at $200 (d512) and this is added to $D300 (d54016) to give the absolute position. Thus we can POKE to this position to blank the cursor and back one to replace the errant character with the cursor.

```
10 PRINT"**********************
15 PRINT
20 PRINT"TYPING PROGRAM
25 PRINT
30 PRINT"**********************
35 PRINT
40 PRINT:PRINT"PRESS RUB OUT OR
    SHIFT 'O' TO ERASE
50 PRINT:PRINT"PRESS SHIFT RUBOUT
    TO CLEAR SCREEN
60 PRINT:PRINT"PRESS ESC TO ESCAPE
    BACK TO BASIC
70 PRINT:PRINT
100 POKE11,0:POKE12,253
110 X=USR(X)
120 P=PEEK(531)
130 IFP=27THEN200
140 IFP=13THENPRINTCHR$(10);
150 IFP=127ORP=95THEN300
160 IFP=111THEN400
170 PRINTCHR$(P);
180 GOTO 110
200 END
300 S=PEEK(512):S1=54016
```

```
 310 POKE S1+S,32:S=S-1
 320 POKE S1+S,95
 330 POKE512,S
 340 GOTO110
 400 FORT=0TO25:PRINT:NEXT
 410 GOTO110
OK
```

Hard Disk Problems

Jim Cressie of Compustat, Inc., in Chicago, has a cautious compliment quickly followed by a caveat:

"Generally speaking maintenance service available through the OSI distributor is excellent, with one glaring exception: maintenance service for the OKIDATA 3306 hard disk. Service costs and turn around time on most other components have been both reasonable and acceptable.

"In May, 1980, my 3308 failed and the fault was in the servo board. I was told that the unit had to be sent back to OKIDATA in California for repair because the distributor had neither the test equipment nor the technical data to repair the board. I was also told that such repair could take six to eight weeks to complete. Cost estimates ranged between $250 and $750 depending on what else might be found, such as a bad disk surface.

"The 3306 was not returned until the middle of August and the cost for repair was over $2000. And it was still broke! And it had to go back to OKIDATA again! And it wasn't returned to the distributor until November!"

As OSIO members were among the first to learn (from the December Newsletter), OSI was buying the California OKIDATA hard disk factory at about the time Jim was climbing the walls and waiting. Thus, for the 74MB hard disk, the responsibilities are no longer spread among the component manufacturer, OSI, the dealer, and the buyer -- instead of a four-way balance, we now have only three. Granted, most of us today don't have hard disks -- but it has been only about five years since the really advanced hobbyist boasted that he could save his programs on paper tape. Hard 5-1/4" disks are now showing up among advanced workers, so -- even if you're using a cassette today, you may adding a hard disk to your system later.

And when you do, there are several things you'll want to know: What's the mean time between failures? What kinds of repairs can be made locally, and what kinds send the equipment back to the factory? Is loaner or rental equipment available while it's gone? And most important of all, what arrangements have you made to make sure that your data remain available even if the equipment dies?

Of course, the same considerations apply to other equipment, too; the big difference is that most other components are pretty well understood by competent dealers, and technical data are available. A hard disk system is likely to have parts that are unfamiliar to even a topnotch computer specialist, and parts that require highly specialized testing and calibration equipment.

We feel sure that this subject is worthy of a textbook -- maybe more than one. What responsibilities should the buyer assume? The dealer? The manufacturer? How do you get these responsibilities spelled out and agreed to? How do you find out quickly what's wrong? (We recall an earlier plea for diagnostics for the hard disk.) How does the user decide what files need to be backed up and when? What's the best way to go about file backup?

We aren't about to publish all those textbooks, but we'd like very much to have your ideas and opinions. One thing we're sure of is this: Some day you're going to flip the switch and get either silence or blue smoke. If you've made the proper preparations, that'll be a great nuisance, and maybe a costly one. If you have not prepared for it, and a lot of vital data is forever gone, it could be a tragedy.

What are your views and suggestions?

Meetings

The MD/VA/DC group meets on the first Tuesday of each month at 7:30 p.m., at the Walter Johnson High School, 6400 Rock Spring Drive, Rockville, MD.

On May 5, Bill Collier will discuss the OSI implementation of CP/M, and Walter

continued on page 8

Scott will explain the DMS Planner. There will also be the usual wide-ranging discussion of hardware and software problems and solutions for all systems.

The OSI Users Group of Southern Ontario will hold its quarterly meeting in the General Sciences Building of McMaster University at Hamilton, 10 a.m. to 4 p.m. on June 6.

OSI discussion groups meet on the second, third, and fourth Tuesday of each month at Community Computers, 2704 N. Pershing Drive, Arlington, VA. On May 12, information will be shared on interfacing and programming printers to operate with the Superboard, C1P, and C4P. On the second Tuesday of succeeding months, discussions will center on cassette-based systems, BASIC-in-ROM, and hardware.

On the third Tuesday of each month the subject will be OS-65D, home systems, polled keyboards, and MDMS. On the fourth Tuesday discussions will be about 65U, serial terminals, 8" floppy and hard disks, and business applications.

OSIO Officers

President:
John Randall, 10220 Conover Drive,
Silver Spring, MD 20850
Executive Vice-President:
David Morganstein, 13329 Woodruff Place,
Germantown, MD 20767
Treasurer and Corporate Secretary:
Mary Lowery, 2005 N. Uhle St.,
Arlington, VA 22201
Corresponding Secretary and
Newsletter Editor: Wallace Kendall,
9002 Dunloggin Rd.,
Ellicott City, MD 21043
Directors:
Sidney H. Brounstein, 9612 McAlpine Rd.,
Silver Spring, MD 20901
William J. Callaghan, 6605 Fisher Ave.,
Falls Church, VA 22046
Joe Kirshner, 9234 E. Parkhill Dr.,
Bethesda, MD 20034
G. D. Marton, 2710 S. Joyce St.,
Arlington, VA 22202
C. H. Popenoe, 6307 Wiscasset Rd.,
Washington, DC 20016
Thomas M. Rowlett, 11296 Windsor Ct.,
Ijamsville, MD 21754

OSIO
MATRIX
6845 Elm Street, Suite 107
McLean, Virginia 22101

BULK RATE
U. S. POSTAGE
PAID
McLEAN, VIRGINIA
PERMIT NO. 178

contains time sensitive material